UNITED STATES PATENT AND TRADEMARK OFFICE

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/061,384	01/31/2002	Khoa Nguyen	SP-6823 US	2421

24209	7590	01/14/2005

GUNNISON MCKAY & HODGSON, LLP
1900 GARDEN ROAD
SUITE 220
MONTEREY, CA 93940

EXAMINER
STEELMAN, MARY J

ART UNIT	PAPER NUMBER
2122	

DATE MAILED: 01/14/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

	Application No.	Applicant(s)
Office Action Summary	10/061,384	NGUYEN ET AL.
	Examiner	Art Unit	
	Mary J. Steelman	2122	

-- *The MAILING DATE of this c mmunication appears on the cover sheet with the correspondence address* --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.
- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1) ☒ Responsive to communication(s) filed on *1/31/02, 4/23/02, 11/24/03*.
2a) ☐ This action is **FINAL**.   2b) ☒ This action is non-final.
3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4) ☒ Claim(s) *1-27* is/are pending in the application.
    4a) Of the above claim(s) _____ is/are withdrawn from consideration.
5) ☐ Claim(s) _____ is/are allowed.
6) ☒ Claim(s) *1-27* is/are rejected.
7) ☐ Claim(s) _____ is/are objected to.
8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9) ☐ The specification is objected to by the Examiner.
10) ☒ The drawing(s) filed on *31 January 2002* is/are: a) ☐ accepted or b) ☒ objected to by the Examiner.
    Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
    Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
    a) ☐ All b) ☐ Some * c) ☐ None of:
      1. ☐ Certified copies of the priority documents have been received.
      2. ☐ Certified copies of the priority documents have been received in Application No. _____.
      3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).
    * See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)
1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
    Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413)
    Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1.   Claims 1-27 are pending.

Drawings

2.   The drawings are objected to because:

FIG. 4, #430, recites "Compute stride stride information…", should be –Compute stride

information…--

FIG. 4, #440 recites, "…the stide pattern…, should be --…the stride pattern…--

Corrected drawing sheets in compliance with 37 CFR 1.121(d) are required in reply to the Office

action to avoid abandonment of the application. Any amended replacement drawing sheet should

include all of the figures appearing on the immediate prior version of the sheet, even if only one

figure is being amended. The figure or figure number of an amended drawing should not be

labeled as "amended." If a drawing figure is to be canceled, the appropriate figure must be

removed from the replacement sheet, and where necessary, the remaining figures must be

renumbered and appropriate changes made to the brief description of the several views of the

drawings for consistency. Additional replacement sheets may be necessary to show the

renumbering of the remaining figures. **The replacement sheet(s) should be labeled**

"Replacement Sheet" in the page header (as per 37 CFR 1.84(c)) so as not to obstruct any

portion of the drawing figures. If the changes are not accepted by the examiner, the applicant will

be notified and informed of any required corrective action in the next Office action. The

objection to the drawings will not be held in abeyance.

Claim Rejections - 35 USC § 103

3.   The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all

obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in
> section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are
> such that the subject matter as a whole would have been obvious at the time the invention was made to a person
> having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the
> manner in which the invention was made.

4.   Claims 1-27 are rejected under 35 U.S.C. 103(a) as being unpatentable over Pre Grant

Pub 2002/0199178 A1 to Hobbs et al.

1, 26, and 27:

-method ...computer-readable medium...system...

([0010], "...method...", [0020], "...encoded on a variety of program storage media...(computer-

readable medium)", [0019], "...computer that may be used to implement the instant

invention...(system)")

-collecting information describing a frequency of occurrence of a plurality of cache

misses caused by at least one instruction;

([0066], "...analyze the program to determine where cache misses and/or cache thrashing is

occurring, and then apply the optimization...")

-identifying a performance degrading instruction;

([0067], "If execution profile data contains information about which loop bodies contain

instructions that are encountering cache misses...(identify degrading instruction)")

-optimizing the program to provide an optimized sequence of instructions, the

optimized sequence of instructions comprising at least one prefetch instruction;

([0066], "...then apply the optimization techniques...")

-modifying the program being executed to include the optimized sequence.

([0067], "...profile data can be used to direct which loops are restructured...")

Hobbs disclosed an invention that [0002] "restructures program loops to reduce cache thrashing.

Hobbs failed to provide specific details related to 'frequency of occurrence of a plurality of

cache misses', but did broadly disclose that analysis was performed (including analysis of

caches) and used in the restructuring of code. Therefore, it would have been obvious, to one of

ordinary skill in the art, at the time of the invention, to modify Hobbs invention to specifically

disclose that 'frequency of occurrence of a plurality of cache misses' was a consideration in the

analysis, because he specifically discussed techniques to minimize cache thrashing caused by

cache misses at [0025-0033].

2.

-the program comprises a plurality of sequence of instructions.

([0010], "...identifying a loop in a program (sequence of instructions)...")

3.

-the performance degrading instruction contributes to highest frequency of occurrence of the

plurality cache misses.

([0022], "...The process then generates execution profile data that represents the results of its

observations and analysis of execution of the code...", [0029], "The organization of the caches

may contribute to or cause thrashing (cache misses)." Broadly Hobbs disclosed 'frequency of

occurrence' by disclosing that an analysis is performed on the execution of the code. Cache

misses are considered in the analysis.)

4.

-the performance degrading instruction contributes to highest degradation in the program

performance.

(([0022], "...The process then generates execution profile data that represents the results of its

observations and analysis of execution (program degradation) of the code...", Broadly Hobbs

disclosed 'highest degradation of performance' by disclosing that an analysis is performed on the

execution of the code. Degradation is considered in the analysis.)

Hobbs disclosed an invention that [0002] "restructures program loops to reduce cache thrashing.

Hobbs failed to provide specific details related to 'highest degradation', but did broadly disclose

that analysis was performed (for the purpose of finding instruction sequences that degrade

program performance) and used in the restructuring of code. Therefore, it would have been

obvious, to one of ordinary skill in the art, at the time of the invention, to modify Hobbs

invention to specifically disclose that 'highest degradation' was a consideration in the analysis,

because he specifically discussed techniques to minimize degrading instructions ([0051]

...optimized u8sing...cache management instructions...").

5.

-wherein the at least one instruction is the performance degrading instruction.

([0002], "The present invention relates generally to software compilers and, more particularly, to a compiler that restructures program loops to reduce cache thrashing." The invention finds degrading instructions in a program.)

6.

-optimizing the program comprises inserting the at least one prefetch instruction prior to the performance degrading instruction.

([0021], "...The optimization processes may perform this transformation of the code...", [0033], "...the compiler operates to insert prefetch...")

7.

-the plurality cache misses are L2/L3 cache misses.

([0025-0027], "Two-level caches...", "...if the requested data is not present in the first cache (a miss), then the request is passed to the second cache...", [0029], "The organization of the caches may contribute to or cause thrashing (cache misses)..." Hobbs disclosed the L1 and L2 caches and considers the cache misses as a potential degrading opportunity.)

8.

-the optimized sequence is prepared while the program is placed in a suspend mode.

([0022], "...the process generates data by initially executing the code...observing and analyzing...The process then generates execution profile data...The optimization process may then use the data to generate a new more efficient version (in a suspend mode)...which is then

processed...The code generator process translates the expanded intermediate code into

instructions that are specific to the architecture...the generator modifies the code...execute (in a

resume mode / change to an execution mode) ...")

9.

-modifying the program comprises:

changing the program from the suspend mode to the execution mode.

See rejection of limitations in claim 8 above.

10.

-optimizing the program comprises:

-receiving information describing a dependency graph for the at least one instruction;

([0036], "...performing use dependence analysis...")

-determining whether a cyclic dependency pattern exists in the dependency graph;

([0036], "...dependence analysis will identify memory operations that have some dependency

upon one another...If there is a...cyclic dependency cycle...")

-if the cyclic dependency pattern exists then, computing stride information derived from the

cyclic dependency pattern;

([0037], "It is envisioned that loop restructuring may be advantageously performed on memory

arrays with longer strides...length of stride in an array that will benefit from loop

restructuring..." Hobbs gave consideration to the stride information when restructuring loop instructions.)

-inserting the prefetch instruction derived from the stride information, the prefetch instruction being inserted into the program prior to the performance degrading instruction. (0041], "...by including features such as ...prefetch instructions...", [0051], "The computer may include a variety of prefetch...instructions that can migrate data..." Additionally see sample code in [0052] that incorporates prefetch instructions.)

11.

-the dependency graph is a backward slice from the performance degrading instruction. ([0036], "...dependence analysis will identify memory operations that have some dependency upon one another...bi-directional dependency...uni-directional dependency...." Hobbs gives consideration to modifying code by understanding the dependency and producing code that preserves the dependency.)

12.

-modifying the program comprises:

-storing the optimized sequence;

([0019], "The computer may be used to compile a source program...to execute (a stored compiled program) a target program that has been compiled...", [0023], The object code is then generated...linker the combines...to produce machine dependent code that is executable...to

generate the execution profile data...used by the processes ...if the code does not exhibit optimal

execution performance...more optimal versions of the code may be generated." Hobbs

generates optimized code, and stores it in preparation for execution.)

-redirecting a sequence of instructions having the performance degrading instruction to include

the optimized sequence.

(As an example, see the code samples in [0052] which redirects a sequence of instructions, by

using prefetch instructions.)

13.

A method of optimizing a program comprising a plurality of execution paths, the method

comprising:

-collecting information describing a plurality of occurrences of a plurality of cache miss events

during a runtime mode of the program;

([0022], "...observing and analyzing the manner in which the code uses resources (collecting

information)..." Cache miss events are analyzed in the runtime generation of profile data. See

more details regarding cache misses at [0025-0033].)

-identifying a performance degrading execution path in the program;

([0022], "...then generates execution profile data that represents the results of its observations

and analysis of execution of the code (performance degrading identified).")

-modifying the performance degrading execution path to define an optimized execution path, the

optimized execution path comprising at least one prefetch instruction;

([0022], "...then use the data to generate (modify) a new more efficient version...", [0033],

"...restructuring the loop...insert (modifying) prefetch and other cache management

instructions...")

-storing the optimized execution path;

([0022], "...the generator modifies the code such that the code reflects scheduling and other low-

level optimizations of the code which are dependent on the architecture of the computer that will

execute the code (modified code is stored and ready for execution)."

-redirecting the performance degrading execution path in the program to include the optimized

execution path.

([0022], "...the generator modifies (redirects) the code such that the code reflects scheduling and

other low-level optimizations of the code which are dependent on the architecture of the

computer that will execute the code (redirected optimized code is stored and ready for

execution).")

Hobbs disclosed an invention that [0002] "restructures program loops to reduce cache thrashing.

Hobbs failed to provide specific details related to 'frequency of occurrence of a plurality of

cache misses', but did broadly disclose that analysis was performed (including analysis of

caches) and used in the restructuring of code. Therefore, it would have been obvious, to one of

ordinary skill in the art, at the time of the invention, to modify Hobbs invention to specifically

disclose that 'frequency of occurrence of a plurality of cache misses' was a consideration in the

analysis, because he specifically discussed techniques to minimize cache thrashing caused by

cache misses at [0025-0033].

14.

-the plurality of cache miss events are caused by an execution of a plurality of performance

degrading instructions.

([0029], "The organization of the caches may contribute or cause thrashing (cache misses).")

15.

-identifying the performance degrading path comprises identifying a performance degrading

instruction contributing to highest plurality of occurrences of cache miss events.

([0022], "...observing and analyzing the manner in which the code uses resources...generates

execution profile data that represents the results of...analysis (identify performance degrading

path and highest plurality of occurrences of cache miss events)...", [0033], "...the instant

invention can also remove the cache thrashing...by restructuring the loop...Additionally, the

compiler operates to insert prefetch and other cache management instructions..." Hobbs

disclosed that cache miss events may contribute to program degradation. See [0025-0033].)

16-18:

-the optimized execution path is defined while placing the program in a suspend mode from the

runtime mode.

-the optimized execution path is executed on resuming the runtime mode of the program code

from the suspend mode.

-redirecting the performance degrading execution path comprises:

-changing the program mode from the suspend mode to the execution mode.

(See limitations as addressed in claims 8 and 9 above.)

19.

-the performance degrading execution path comprises a performance degrading instruction

causing the cache miss event.

(See limitations as addressed in claim 3 above.)

20.

-the at least one prefetch instruction is inserted prior to the performance degrading instruction.

(See limitations as addressed in claim 6 above.)

21.

-identifying the performance degrading execution path comprises determining whether a cache

miss event of the plurality of cache miss events is an L2/L3 cache miss.

(See limitations as addressed in claim 7 above.)

22.

-identifying the performance degrading path comprises identifying a performance degrading

instruction contributing to highest degradation in the program performance.

(See limitations as addressed in claim 4 above.)

23.

-modifying the performance degrading execution path comprises:

-receiving information describing a dependency graph for a program degrading instruction

contributing to highest occurrence of the plurality of cache miss events, the performance

degrading instruction being included in the performance degrading execution path;

([0036], "The loop restructuring routine begins…by performing use dependence analysis…and

how the loop…may be distributed…dependence analysis will identify memory operations (cache

miss events)…")

-determining whether a cyclic dependency pattern exists in the graph;

([0036], "…dependence analysis will identify memory operations that have some

dependency…If there is…cyclic dependency… (determine whether a cyclic dependency pattern

exists)")

-if the cyclic dependency pattern exists then, computing stride information derived from the

cyclic dependency pattern;

([0037], "...loop restructuring may be advantageously performed on memory arrays with longer strides...The particular organization and construction of the cache memory in the computer on which the compiled program is to be executed will influence the length of stride (stride information is considered) in an array that will benefit from loop restructuring.")

-inserting the at least one prefetch instruction derived from the stride information, the at least one prefetch instruction being inserted into the optimized execution path prior to the performance degrading instruction.

([0033], "...compiler operates to insert prefetch..." Also see [0051].

Also, note limitations as addressed in claim 10 above.)

24.

-the dependency graph is a backward slice from the performance degrading instruction.

(See limitations as addressed in claim 11 above.)

25.

A method of optimizing a program, the method comprising:

-receiving information describing a dependency graph for an instruction causing frequent cache misses, the instruction being included in the program;

([0036], "...performing use dependence analysis...")

-determining whether a cyclic dependency pattern exists in the graph;

([0036], "If there is ...a cyclic dependency...")

-if the cyclic dependency pattern exists then, computing stride information derived from the

cyclic dependency pattern;

([0037], "...length of stride in an array that will benefit from loop restructuring.")

-inserting an at least one prefetch instruction derived from the stride information, the at

least one prefetch instruction being inserted into the program prior to the instruction causing the

frequent cache misses;

([0051], "...the cache may be further optimized using, for example, cache management

instructions. The computer may include a variety of prefetch and cache management instructions

...")

-reusing the at least one prefetch instruction in the program for reducing subsequent cache

misses;

([0023], "...the executable code is executed by the process to generate the execution profile data

that is used by the processes to determine whether the code, when executed, exhibits optimal

execution performance, and if the code does not exhibit optimal execution performance, may be

used by the processes to make more optimal versions of the code from which more optimal

versions of the code may be generated (reuse for reducing subsequent cache misses)."

-performing said receiving, said determining, said computing, said inserting and said reusing

during runtime of the program.

([0023], """...the executable code is executed (runtime) by the process to generate the execution

profile data that is used by the processes to determine whether the code, when executed (runtime)

, exhibits optimal execution performance...")

Also, see limitations addressed in claim 23 above.

Hobbs disclosed an invention that [0002] "restructures program loops to reduce cache thrashing.

Hobbs failed to provide specific details related to 'frequency of occurrence of a plurality of

cache misses', but did broadly disclose that analysis was performed (including analysis of

caches) and used in the restructuring of code. Therefore, it would have been obvious, to one of

ordinary skill in the art, at the time of the invention, to modify Hobbs invention to specifically

disclose that 'frequency of occurrence of a plurality of cache misses' was a consideration in the

analysis, because he specifically discussed techniques to minimize cache thrashing caused by

cache misses at [0025-0033].

Conclusion

5.   The prior art made of record and not relied upon is considered pertinent to applicant's

disclosure.

Any inquiry concerning this communication or earlier communications from the

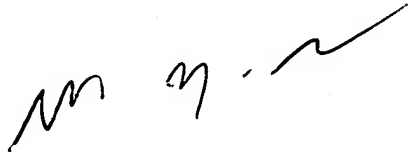examiner should be directed to Mary Steelman, whose telephone number is (571) 272-3704. The

examiner can normally be reached Monday through Thursday, from 7:00 AM to 5:30 PM   If

attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan

Q. Dam can be reached at (571) 272-3695. The fax phone number for the organization where

this application or proceeding is assigned is 703-872-9306.

Information regarding the status of an application may be obtained from the Patent

Application Information Retrieval (PAIR) system. Status information for published applications

may be obtained from either Private PAIR or Public PAIR. Status information for unpublished

applications is available through Private PAIR only. For more information about the PAIR

system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR

system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Mary Steelman

01/06/2005

WEI Y. ZHEN
PRIMARY EXAMINER